b. Depressive Disorder

```
# creating a regression object

regression_depressive = LinearRegression()
```

```
# fitting the regression object onto the training set

regression_depressive.fit(X_train, y_depressive_train)
```

```
LinearRegression()
```

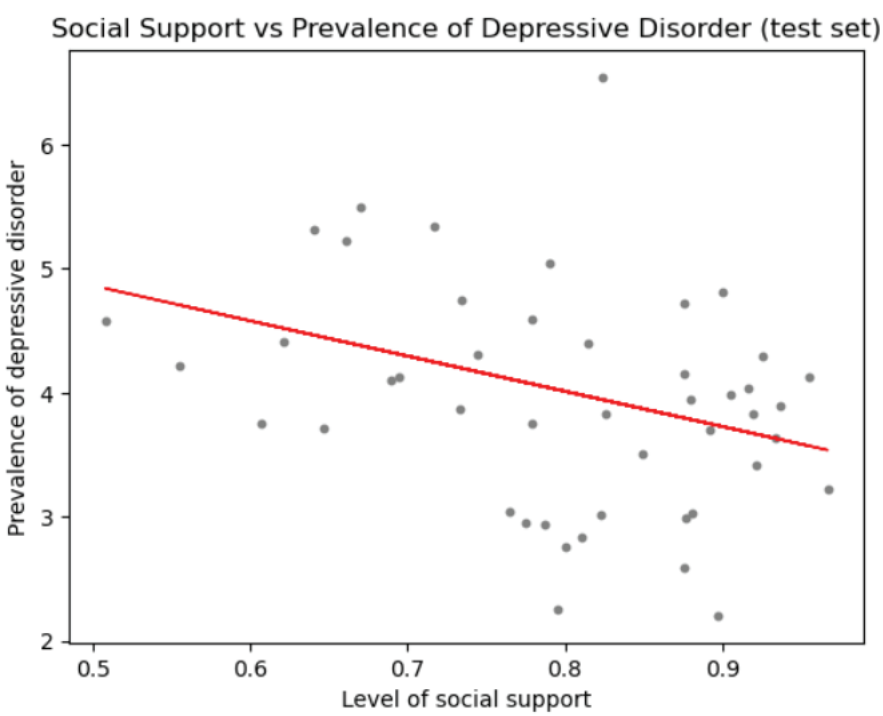
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
# predicting the values of y using X

y_depressive_predicted = regression_depressive.predict(X_test)
```

```
# creating a scatterplot that shows the regression line from the model on the test set

plot_test = plt
plot_test.scatter(X_test, y_depressive_test, color='gray', s = 10)
plot_test.plot(X_test, y_depressive_predicted, color='red', linewidth =1)
plot_test.title('Social Support vs Prevalence of Depressive Disorder (test set)')
plot_test.xlabel('Level of social support')
plot_test.ylabel('Prevalence of depressive disorder')
plot_test.show()
```




```
# creating objects that contain the model summary statistics

rmse_depressive = mean_squared_error(y_test, y_depressive_predicted)
r2_depressive = r2_score(y_test, y_depressive_predicted)
```

```
# printing the model summary statistics to evaluate performance of model

print('Slope:' ,regression_depressive.coef_)
print('Mean squared error: ', rmse_depressive)
print('R2 score: ', r2_depressive)
```

```
Slope: [[-2.84020461]]
Mean squared error:  2.9177347863543024
R2 score:  -0.4314165528979834
```

```
# creating a dataframe comparing actual and predicted values of y

data = pd.DataFrame({'Actual': y_depressive_test.flatten(), 'Predicted': y_depressive_predicted.flatten()})
data.head(30)
```